RP HW

Problem1 - Obtain and plot the values of $R_{ss}[m], \forall 0 \leq m \leq 20$, in Figure 1.

By Yule-Walker equations , We can get

$$
\begin{aligned}
&R_{ss}[0] + a_1 R_{ss}[1] a_2 R_{ss}[2] + a_3 R_{ss}[3] + \cdots + a_{20} R_{ss}[20] = b_0^2 \\
&R_{ss}[1] + a_1 R_{ss}[0] a_2 R_{ss}[1] + a_3 R_{ss}[2] + \cdots + a_{19} R_{ss}[20] = 0 \\
&\vdots \\
&R_{ss}[20] + a_1 R_{ss}[19] a_2 R_{ss}[18] + a_3 R_{ss}[17] + \cdots + a_{20} R_{ss}[0] = 0
\end{aligned}
$$

Calculate and plot the values of $R_{ss}[m]$ for $0 \leq m \leq 20$ by substituting the values of $a_1, a_2, \ldots, a_{20}$, and b_0 into the equation.

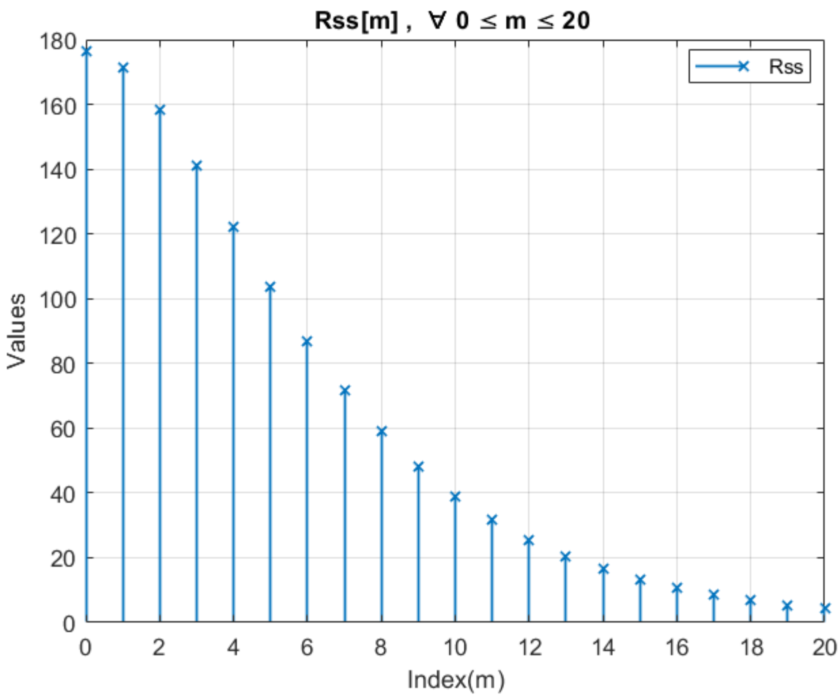

Figure 1: Rss

```
noiseLength = 1000000; % length s[n],n=[1:1000000]
s=zeros([1 noiseLength+3]); % s inital value
a=[1,-1.9,1.18,-0.24]; % mean a0,a1,a2,a3
b0=2;
% set white Gaussian noise value

meanValue = 0;      % mean value
whiteNoise = meanValue + randn(1, noiseLength);
varianceValue = var(whiteNoise);
disp('White Noise Variance :');
disp(varianceValue);

% find s[n] value
% Note matlab idx from 1 not 0
for i=4:noiseLength+3
    s(i)=b0*whiteNoise(i-3)-a(2)*s(i-1)-a(3)*s(i-2)-a(4)*s(i-3);
end
s=s(4:end); % remove init point

M=20;
% slove 4-order equation at first
% Ax=b find x , where x is Rss[0]~Rss[3]
A=[a(1),a(2),a(3),a(4);
   a(2),a(1)+a(3),a(4),0;
   a(3),a(2)+a(4),a(1),0;
   a(4),a(3),a(2),a(1)];
b=[b0^2;0;0;0];
Rss=transpose(A\b);
disp('Rss[0] ~ Rss[3]:');
disp(Rss);
Rss=[Rss,zeros([1,17])];
% slove Rss[4]~Rss[20]
for i=5:M+1
    % Rss[4(i)]=-a1*Rss[3(i-1)]-a2*Rss[2(i-2)]-a3*Rss[1(i-3)]
    Rss(i)=-a(2)*Rss(i-1)-a(3)*Rss(i-2)-a(4)*Rss(i-3);
end
```

Listing 1: Problem1 Matlab Code

Problem2 - plot $\{\hat{s}_{20}[n], \forall 10000 \leq n \leq 10100\}$ and $\{s[n], \forall 10000 \leq n \leq 10100\}$, in Figure 2
According to the definition of $\hat{s}_M[n] = \hat{E}\{s[n] \mid s[n-k], 1 \leq k \leq M\}$ and substituting M = 20 , $\forall 10000 \leq n \leq 10100$ into the definition , we can get

$$\hat{s}_{20}[n] = \hat{E}\{s[n] \mid s[n-k], 1 \leq k \leq 20\}$$

Based on theorem 13-1 , we know that $E[(s[n] - \sum_{k=1}^{N} h[k]s[n-k])s[n-m]] = 0, \forall 1 \leq m \leq 20$ .According to the result obtained from the previous equation , We can determine the value of $h[k], where \forall 1 \leq k \leq 20$,by solving the following system of simultaneous equations :

$$R_{ss}[1] - \sum_{k=1}^{N} h[k]R_{ss}[1-k] = 0$$
$$R_{ss}[2] - \sum_{k=1}^{N} h[k]R_{ss}[2-k] = 0$$
$$\vdots$$
$$R_{ss}[20] - \sum_{k=1}^{N} h[k]R_{ss}[20-k] = 0$$

Finally ,we can calculate the $\hat{s}_{20}[n] = \sum_{k=1}^{N} h[k]s[n-k], \forall 10000 \leq n \leq 10100$

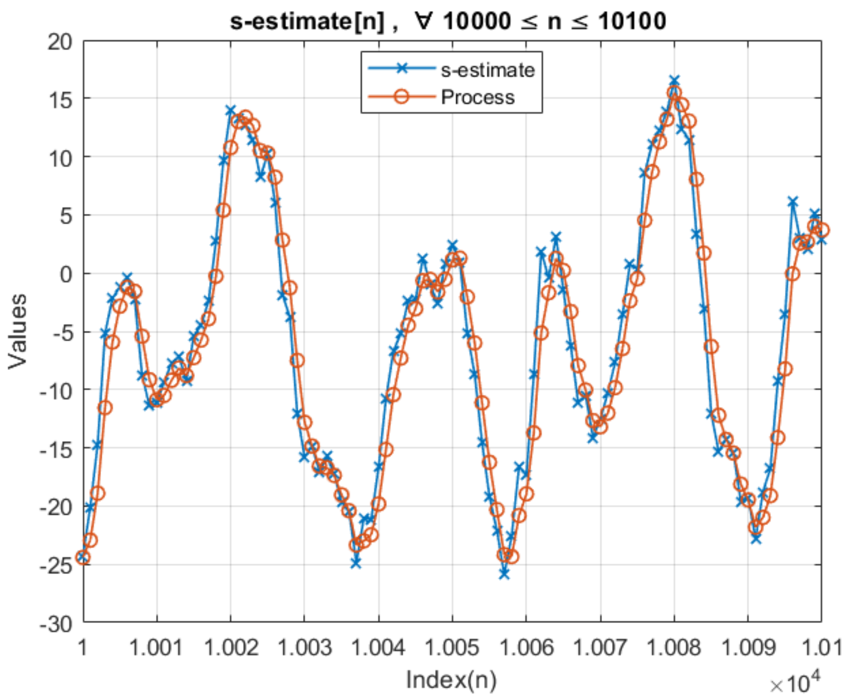

Figure 2: $\hat{s}_{20}[n], \forall 10000 \leq n \leq 10100$

```
b=transpose(Rss(2:21));
A=[];
for i=1:M
    tmp=flip(Rss(1:i));
    if M-i>0
        tmp=[tmp,Rss(2:M-i+1)];
    end
    A=[A;tmp];
end

if transpose(A)==A
    disp('Problem2 - A matrix is same');
end
h=transpose(A\b);

s_estimate = [];
for i=n
    tmp=h.*flip(s(i-M+1:i));
    s_estimate=[s_estimate,sum(tmp)];
end
```

Listing 2: Problem2 Matlab Code

Problem3 - $\hat{P}_{20}[n], \forall n \in \{10, 10^2, 10^3, 10^4, 10^5, 10^6\}$ in Figure 3
Based on definition $\hat{P}_M[n] = \frac{1}{n}\sum_{k=1}^{n}(\hat{s}_M[k] - s[k])^2$, we can directly calculate the error between predicted values and actual values.

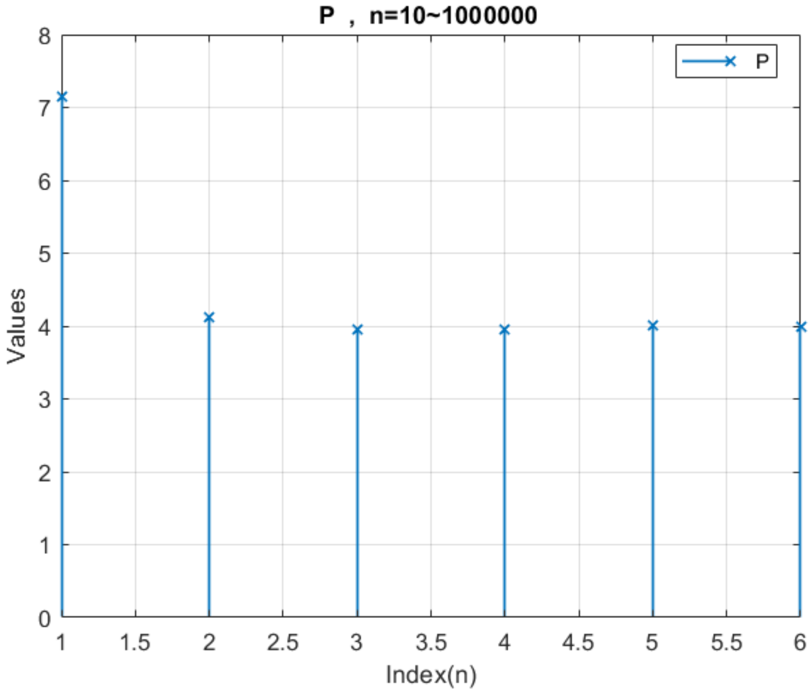

Figure 3: $\hat{P}_{20}[n], \forall n \in \{10, 10^2, 10^3, 10^4, 10^5, 10^6\}$

```matlab
%% Count all s_estimate
s_estimate=zeros([1 noiseLength]); % clean all
for n=1:M
    tmp=0;
    for k=1:M
        if n-k>0
            tmp=tmp+h(k)*s(n-k);
        end
    end
    s_estimate(n)=tmp;
end

for n=M+1:noiseLength
    s_estimate(n)=sum(h.*flip(s(n-M:n-1)));
end
%%  Problem-3  n=10-1
p_10=0;
n_10=10;
for i=1:n_10
    tmp=(s_estimate(i)-s(i))^2;
    p_10=p_10+tmp;
end
p_10=p_10/10;
d
%%  Problem-3  n=100-2
p_100=0;
n_100=100;
for i=n_10+1:n_100
    tmp=(s_estimate(i)-s(i))^2;
    p_100=p_100+tmp;
end
p_100=((p_10*n_10) + p_100)/n_100;

%%  Problem-3  n=1000-3
p_1000=0;
n_1000=1000;
for i=n_100+1:n_1000
    tmp=(s_estimate(i)-s(i))^2;
    p_1000=p_1000+tmp;
end
p_1000=((p_100*n_100) + p_1000)/n_1000;

%%  Problem-3  n=10000-4
p_10000=0;
n_10000=10000;
for i=n_1000+1:n_10000
    tmp=(s_estimate(i)-s(i))^2;
    p_10000=p_10000+tmp;
end
p_10000=((p_1000*n_1000) + p_10000)/n_10000;
%%  Problem-3  n=100000-5
p_100000=0;
n_100000=100000;
for i=n_10000+1:n_100000
    tmp=(s_estimate(i)-s(i))^2;
    p_100000=p_100000+tmp;
end
p_100000=((p_10000*n_10000) + p_100000)/n_100000;
%%  Problem-3  n=1000000-6
p_1000000=0;
n_1000000=1000000;
for i=n_100000+1:n_1000000
    tmp=(s_estimate(i)-s(i))^2;
    p_1000000=p_1000000+tmp;
end
p_1000000=((p_100000*n_100000) + p_1000000)/n_1000000;
```

Listing 3: Problem3 Matlab Code

Problem4 - Consider the case in which $\{i[n], -\infty < n < \infty\}$ is a white noise process but $i[n]$ has the following probability density function for each $n \in Z$.

$$f_{i[n]}(t) = \begin{cases} 0.5e^{-t}, \forall t > 0 \\ 0.5e^{t}, \forall t < 0 \end{cases}$$

Plot $\hat{P}_{20}[n], \forall n \in \{10, 10^2, 10^3, 10^4, 10^5, 10^6\}$, in Figure 4
Based on definition $\hat{P}_M[n] = \frac{1}{n}\sum_{k=1}^{n}(\hat{s}_M[k] - s[k])^2$, we can directly calculate the error between predicted values and actual values.

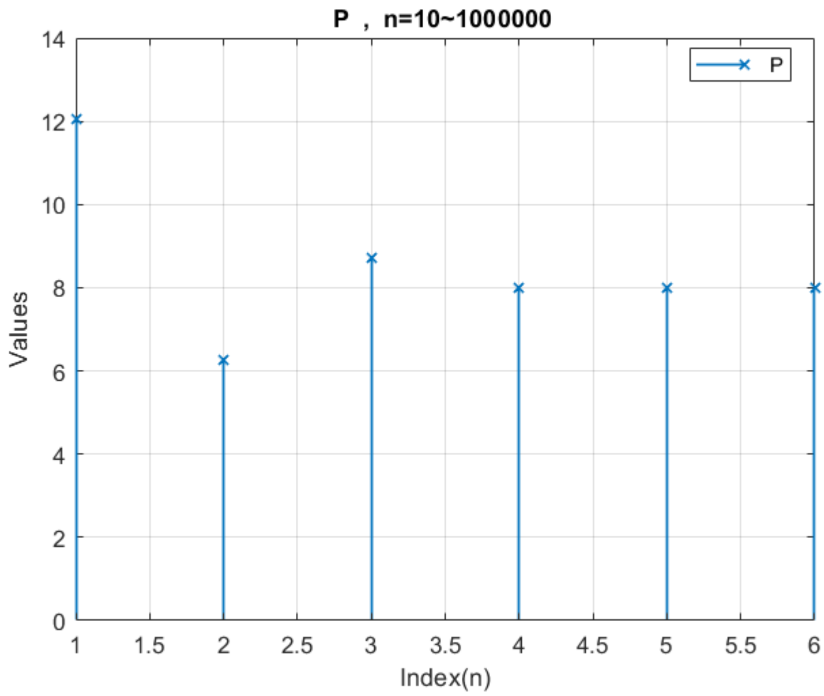

Figure 4: $\hat{P}_{20}[n], \forall n \in \{10, 10^2, 10^3, 10^4, 10^5, 10^6\}$

```
%% create Laplace distribution noise
noiseLength=1000000;
s=zeros([1 noiseLength+3]); % s inital value
a=[1,-1.9,1.18,-0.24]; % mean a0,a1,a2,a3
b0=2;

m=1;
mu = 0;
sigma = sqrt(2);
u = rand(m, noiseLength)-0.5;
b = sigma / sqrt(2);
laplaceRandomNoise = mu - b * sign(u).* log(1- 2* abs(u));

for i=4:noiseLength+3
    s(i)=b0*laplaceRandomNoise(i-3)-a(2)*s(i-1)-a(3)*s(i-2)-a(4)*s(i-3);
end

s=s(4:end); % remove init point
```

Listing 4: Problem4 - Create Laplace distribution noise

```
%% Count Rss
M=20;
% Ax=b find x , x is Rss[0]~Rss[3]
A=[a(1),a(2),a(3),a(4);
   a(2),a(1)+a(3),a(4),0;
   a(3),a(2)+a(4),a(1),0;
   a(4),a(3),a(2),a(1)];
b=[b0^2;0;0;0];
Rss=transpose(A\b);
disp('Rss[0] ~ Rss[3]:');
disp(Rss);
Rss=[Rss,zeros([1,17])];
% slove Rss[4]~Rss[20]
for i=5:M+1
    % Rss[4(i)]=-a1*Rss[3(i-1)]-a2*Rss[2(i-2)]-a3*Rss[1(i-3)]
    Rss(i)=-a(2)*Rss(i-1)-a(3)*Rss(i-2)-a(4)*Rss(i-3);
end
```

Listing 5: Problem4 - Count Rss

```
%% Count all s_estimate
s_estimate=zeros([1 noiseLength]); % clean all
for n=1:M
    tmp=0;
    for k=1:M
        if n-k>0
            tmp=tmp+h(k)*s(n-k);
        end
    end
    s_estimate(n)=tmp;
end

for n=M+1:noiseLength
    s_estimate(n)=sum(h.*flip(s(n-M:n-1)));
end
```

Listing 6: Problem4 - Count all $\hat{s}$

```
%% Problem-4  n=10-1
p_10=0;
n_10=10;
for i=1:n_10
    tmp=(s_estimate(i)-s(i))^2;
    p_10=p_10+tmp;
end
p_10=p_10/10;
disp('p_10 :');
disp(p_10);
%% Problem-4  n=100-2
p_100=0;
n_100=100;
for i=n_10+1:n_100
    tmp=(s_estimate(i)-s(i))^2;
    p_100=p_100+tmp;
end
p_100=((p_10*n_10) + p_100)/n_100;
disp('p_100 :');
disp(p_100);
%% Problem-4  n=1000-3
p_1000=0;
n_1000=1000;
for i=n_100+1:n_1000
    tmp=(s_estimate(i)-s(i))^2;
    p_1000=p_1000+tmp;
end
p_1000=((p_100*n_100) + p_1000)/n_1000;
disp('p_1000 :');
disp(p_1000);
%% Problem-4  n=10000-4
p_10000=0;
n_10000=10000;
for i=n_1000+1:n_10000
    tmp=(s_estimate(i)-s(i))^2;
    p_10000=p_10000+tmp;
end
p_10000=((p_1000*n_1000) + p_10000)/n_10000;
disp('p_10000 :');
disp(p_10000);
%% Problem-4  n=100000-5
p_100000=0;
n_100000=100000;
for i=n_10000+1:n_100000
    tmp=(s_estimate(i)-s(i))^2;
    p_100000=p_100000+tmp;
end
p_100000=((p_10000*n_10000) + p_100000)/n_100000;
disp('p_100000 :');
disp(p_100000);
%% Problem-4  n=1000000-6
p_1000000=0;
n_1000000=1000000;
for i=n_100000+1:n_1000000
    tmp=(s_estimate(i)-s(i))^2;
    p_1000000=p_1000000+tmp;
end
p_1000000=((p_100000*n_100000) + p_1000000)/n_1000000;
disp('p_1000000 :');
disp(p_1000000);
```

Listing 7: Problem4 - Count all P